Notes on Practical Reverse Engineering

These are questions I asked myself when I was trying to learn these concepts

1. What are PCR, PRCB and why do we have them?

PCR (Process Control Region) is a per-processor data structure used in the kernel that contains critical information about it. **Inside of it** there's PRCB (Process Region Control Block) which contains other stuff about it (both are undocumented). In order to access the PCR FS/GS are used (x86/x64 respectively).

2. How does an interrupt work ?

The operating system uses data structures (IDT) to know which interrupts are handled and how.

3. What are SYSENTER / SYSEXIT and why do we need them

These two instructions are used to implement syscalls. (The reason we have syscalls in the first place is beacuse user space *can't* directly access hardware / physical mem addresses.)

4. How are syscalls information data stored/retrieved

Windows distinguishes GUI syscalls from non-GUI ones: W32pServiceTable and KiServiceTable. They have a base and a limit.

5. In which way can a system call be implemented ?

Syscalls can be implemented by usign either having a dedicated int code (0x2e) or a trap instructions (SYSENTER/SYSEXIT).

6. Explain what an IRQL is

Interrupt Request Level are numbers associated with interrupts and describe the "priority" of the int. IRQL is an unsigned char and is per-processor: interrupts are handled only if they number is equal or higher than the current IRQL.

doubt 1: How does the CPU know when to increase/decrease IRQL ? Does it have a queue or something ?

7. What are MDLs,why do we need them and what is their relationship w/ the MMU ?

8. Explain execution contests

Unlike Linux, Windows distinguishes threads from processess (every process has at least 1 thread). A contest is a collection of all the information relevant to the process.

doubt 1: Are they 'relevant' for the reverse engineers?

9. Explain what a work item is

Worker threads are kernel mode threads executed on behalf of the caller.